DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

New concave penalty functions for improving the Feasibility Pump

Marianna De Santis
Stefano Lucidi
Francesco Rinaldi

Technical Report n. 10, 2010

New concave penalty functions for improving the Feasibility Pump

M. De Santis, S. Lucidi, F. Rinaldi

Dipartimento di Informatica e Sistemistica
Sapienza Università di Roma
Via Ariosto, 25 - 00185 Roma - Italy

e-mail: mdesantis@dis.uniroma1.it
e-mail: stefano.lucidi@dis.uniroma1.it
e-mail: rinaldi@dis.uniroma1.it

Abstract

Mixed-Integer optimization represents a powerful tool for modeling many optimization problems arising from real-world applications. The Feasibility pump is a heuristic for finding feasible solutions to mixed integer linear problems. In this work, we propose a new feasibility pump approach using concave non-differentiable penalty functions for measuring solution integrality. We present computational results on binary MILP problems from the MIPLIB library showing the effectiveness of our approach.

Keywords. Mixed integer programming, Concave penalty functions, Frank-Wolfe algorithm, Feasibility Pump.

MSC. 90C06, 90C10, 90C11, 90C30, 90C59

1 Introduction

Many real-world problems can be modeled as Mixed Integer Programming (MIP) problems, namely as minimization problems where some (or all) of the variables only assume integer values. Finding a first feasible solution quickly is crucial for solving this class of problems. In fact, many local-search approaches for MIP problems such as Local Branching [13], guide dives and RINS [10] can be used only if a feasible solution is available.
In the literature, several heuristics methods for finding a first feasible solution for a MIP problem have been proposed (see e.g. [3]-[5], [7], [15]-[18], [20], [22]). Recently, Fischetti, Glover and Lodi [12] proposed a new heuristic, the well-known Feasibility Pump, that turned out to be very useful in finding a first feasible solution even when dealing with hard MIP instances. The FP heuristic is implemented in various MIP solvers such as BONMIN [8].
The basic idea of the FP is that of generating two sequences of points $\{\bar{x}^k\}$ and $\{\tilde{x}^k\}$ such that $\bar{x}^k$ is LP-feasible, but may not be integer feasible, and $\tilde{x}^k$ is integer, but not necessarily LP-feasible. To be more specific the algorithm starts with a solution of the LP relaxation $\bar{x}^0$ and sets $\tilde{x}^0$ equal to the rounding of $\bar{x}^0$. Then, at each iteration $\bar{x}^{k+1}$ is chosen as the nearest LP-feasible point in ℓ_1-norm to $\tilde{x}^k$, and $\tilde{x}^{k+1}$ is obtained as the rounding of $\bar{x}^{k+1}$. The aim of the algorithm is to reduce at each iteration the distance between the points of the two sequences, until the two points are the same and an integer feasible solution is found. Unfortunately, it can happen that the distance between $\bar{x}^{k+1}$ and $\tilde{x}^k$ is greater than zero and $\tilde{x}^{k+1} = \tilde{x}^k$, and the strategy can stall. In order to overcome this drawback, random perturbations and restart procedures are performed.
As the algorithm has proved to be effective in practice, various papers devoted to its further improvements have been developed. Fischetti, Bertacco and Lodi [6] extended the ideas on which the FP is based in two different directions: handling MIP problems with both 0-1 and integer variables, and exploiting the FP information to drive a subsequent enumeration phase. In [1], in order to improve the quality of the feasible solution found, Achterberg and Berthold consider an alternative distance function which takes into account the original objective function. In [14], Fischetti and Salvagnin proposed a new rounding heuristic based on a diving-like procedure and constraint propagation.
An interesting interpretation of the FP has been given by J.Eckstein and M.Nediak in [7]. In this work they noticed that the FP heuristic may be seen as a form of Frank-Wolfe procedure applied to a nonsmooth merit function which penalizes the violation of the 0-1 constraints.
In this paper, taking inspiration from [23], we propose new merit functions and we include them in the basic FP scheme [12]. A reported extended computational experience seems to indicate that the use of these new merit functions improves the FP efficiency.
The paper is organized as follows. In Section 2, we give a brief review of the Feasibility Pump heuristic. In Section 3, we show the equivalence between the FP heuristic and the Frank-Wolfe algorithm applied to a nonsmooth merit function. In Section 4, we introduce new nonsmooth merit functions and we discuss their properties. We present our algorithm in Section 5. Computational results are shown in Section 6, where we give a detailed performance comparison of our algorithm with the FP. Some conclusions are drawn in Section 7.

In the following, given a concave function $f : R^n \to R$, we denote by $\partial f(x)$ the set of supergradients of f at the point x, namely

$$\partial f(x) = \{v \in R^n \ : \ f(y) - f(x) \leq v^T(y - x), \ \forall \ y \in R^n\}.$$

2 The Feasibility Pump Heuristic

We consider a MIP problems of the form:

$$\begin{aligned} &\min c^T x \\ &\text{s.t. } Ax \geq b \\ &\quad x_j \in \{0,1\} \ \forall j \in I, \end{aligned} \tag{MIP}$$

where $A \in \mathbf{R}^{m \times n}$ and $I \subset \{1, 2, \ldots, n\}$ is the set of indices of zero-one variables. Let $P = \{x : Ax \geq b\}$ denote the polyhedron of the LP-relaxation of (MIP). The Feasibility Pump starts from the solution of the LP relaxation problem $\bar{x}^0 := \arg\min\{c^T x : x \in P\}$ and generates two sequences of points $\bar{x}^k$ and $\tilde{x}^k$: $\bar{x}^k$ is LP-feasible, but may be integer infeasible; $\tilde{x}^k$ is integer, but not necessarily LP-feasible. At each iteration $\bar{x}^{k+1} \in P$ is the nearest point in ℓ_1-norm to $\tilde{x}^k$:

$$\begin{aligned} \bar{x}^{k+1} := \arg\min \Delta(x, \tilde{x}^k) \\ \text{s.t. } Ax \geq b \end{aligned} \tag{1}$$

where

$$\Delta(x, \tilde{x}^k) = \sum_{j \in I} |x_j - \tilde{x}^k_j|.$$

The point $\tilde{x}^{k+1}$ is obtained as the rounding of $\bar{x}^{k+1}$. The procedure stops if at some index l, $\bar{x}^l$ is integer or, in case of failing, if it reaches a time or iteration limit. In order to avoid stalling issues and loops, the Feasibility Pump performs a perturbation step. Here we report a brief outline of the basic scheme:

The Feasibility Pump (FP) - basic version

Initialization: Set $k = 0$, let $\bar{x}^0 := \arg\min\{c^T x : Ax \geq b\}$
If ($\bar{x}^0$ is integer) return $\bar{x}^0$
Compute $\tilde{x}^0 = round(\bar{x}^0)$
While (not stopping condition) **do**

Step 1 Compute $\bar{x}^{k+1} := \arg\min\{\Delta(x, \tilde{x}^k) : Ax \geq b\}$

Step 2 If ($\bar{x}^{k+1}$ is integer) return $\bar{x}^{k+1}$

Step 3 Compute $\tilde{x}^{k+1} = round(\bar{x}^{k+1})$

Step 4 If (cycle detected) $\tilde{x}^{k+1} = perturb(\tilde{x}^k)$

Step 5 Update $k = k + 1$

End While

Now we give a better description of the rounding and the perturbing procedures used respectively at **Step 3** and at **Step 4**:

Round: This function transforms a given point $\bar{x}^k$ into an integer one, $\tilde{x}^k$. The easiest choice is that of rounding each component $\bar{x}_j^k$ with $j \in I$ to the nearest integer, while leaving the continuous components of the solution unchanged. Formally,

$$\tilde{x}_j^k = \begin{cases} [\bar{x}_j^k] & \text{if } j \in I \\ \\ \bar{x}_j^k & \text{otherwise} \end{cases} \tag{2}$$

where $[\cdot]$ represents scalar rounding to the nearest integer.

Perturb: The aim of the perturbation procedure is to avoid cycling and it consists in two heuristics. To be more specific:

- if $\tilde{x}_j^k = \tilde{x}_j^{k+1}$ for all $j \in I$ a weak perturbation is performed, namely, a random number of integer constrained components, chosen as to minimize the increase in the distance $\Delta(\bar{x}^{k+1}, \tilde{x}^{k+1})$, is flipped.
- If a cycle is detected by comparing the solutions obtained in the last 3 iterations, or in any case after R iterations, a strong random perturbation is performed. For each $j \in I$ a uniformly random value is generated, $\rho_j \in [-0.3, 0.7]$ and if

  $$|\bar{x}_j^{k+1} - \tilde{x}_j^{k+1}| + \max\{\rho_j, 0\} > 0.5$$

  the component $\tilde{x}_j^{k+1}$ is flipped.

Remark 1 *The objective function $\Delta(x, \tilde{x}^k)$ discourages the optimal solution of the relaxation from being "too far" from $\tilde{x}^k$. In practice, the method tries to force a large number of variables of $\bar{x}^{k+1}$ to have the same (integer) value as $\tilde{x}^k$ (see [12]).*

3 The FP heuristic as a Frank-Wolfe algorithm for minimizing a nonsmooth merit function

In a recent work J.Eckstein and M.Nediak [7] noted that the feasibility pump heuristic may be seen as a nonsmooth Frank-Wolfe merit function procedure. In order to better understand this equivalence we recall the unitary stepsize Frank-Wolfe method for concave non-differentiable functions. Let us consider the problem

$$\begin{array}{l} \min f(x) \\ x \in P \end{array} \tag{3}$$

where $P \subset R^n$ is a non empty polyhedral set that does not contain lines going to infinity in both directions, $f : R^n \to R$ is a concave, non-differentiable function, bounded below on P.
The Frank-Wolfe algorithm with unitary stepsize can be described as follows.

Frank-Wolfe - Unitary Stepsize (FW1) Algorithm

Initialization: Set $k = 0$, let $x^0 \in R^n$ be the starting point, compute $g^0 \in \partial f(x^0)$
While $x^k \notin \arg\min_{x \in P} (g^k)^T x$

Step 1 Compute a vertex solution x^{k+1} of

$$\min_{x \in P} (g^k)^T x$$

Step 2 Compute $g^{k+1} \in \partial f(x^{k+1})$, update $k = k + 1$

End While

The algorithm involves only the solution of linear programming problems, and the following result, proved in [24], shows that the algorithm generates a finite sequence and that it terminates at a stationary point $x^\star$, namely a point satisfying the following condition:

$$(g^\star)^T(x - x^\star) \geq 0, \ \ \forall x \in P \tag{4}$$

with $g^\star \in \partial f(x^\star)$.

Proposition 1 *The Frank-Wolfe algorithm with unitary stepsize converges to a vertex stationary point of problem* (3) *in a finite number of iterations.*

Now we consider the basic FP heuristic without any perturbation (i.e. without Step 4) and we show that it can be interpreted as the Frank-Wolfe algorithm with unitary stepsize applied to a concave, nondifferentiable merit function.
First of all, we can easily see that

$$\Delta(x, \tilde{x}^k) = \sum_{j \in I : \tilde{x}^k_j = 0} x_j - \sum_{j \in I : \tilde{x}^k_j = 1} x_j.$$

At each iteration, the Feasibility Pump for mixed 0-1 problems computes, at Step 1, the solution of the LP problem

$$\begin{aligned} \bar{x}^{k+1} \in \arg\min \Delta(x, \tilde{x}^k) \\ \text{s.t.} \ \ Ax \geq b \\ 0 \leq x_j \leq 1 \ \forall j \in I. \end{aligned} \tag{5}$$

Then, at Step 3, it rounds the solution $\bar{x}^k$, thus giving $\tilde{x}^{k+1}$.
These two operations can be included in the unique step of calculating the solution of the following LP problem:

$$\begin{aligned} \min \ & \sum_{j \in I : \bar{x}^k_j < \frac{1}{2}} x_j - \sum_{j \in I : \bar{x}^k_j \geq \frac{1}{2}} x_j \\ \text{s.t.} \ \ & Ax \geq b \\ & 0 \leq x_j \leq 1 \ \forall j \in I, \end{aligned} \tag{6}$$

which can be seen as the iteration of the Frank Wolfe method with unitary stepsize applied to the following minimization problem

$$\begin{aligned} \min & \sum_{i \in I} \min\{x_i, 1 - x_i\} \\ \text{s.t.} & \ Ax \geq b \\ & 0 \leq x_j \leq 1 \ \forall j \in I. \end{aligned} \tag{7}$$

4 New nonsmooth merit functions for the FP approach

As we have seen in the previous section, the basic Feasibility Pump is equivalent to minimizing a separable nonsmooth function which penalizes the 0-1 infeasibility, namely

$$\psi(x) = \sum_{i \in I} \phi(x_i) \tag{8}$$

where $\phi : R \to R$ is a concave nonsmooth function given by

$$\phi(t) = \min\{t, 1 - t\}. \tag{9}$$

Taking into account the Remark 1, we propose new functions ϕ that should improve the ability of the algorithm to force the new LP-feasible solution to have the same (integer) value as the integer feasible solution obtained at the previous iteration. Our idea is to replace the linear terms in ϕ with suitable nonlinear terms that lead to a merit function ψ whose feature is that of encouraging the change of a bunch of variables rather than distributing this change over all the variables.
We start by giving an example in R^2.

Example 1 *Suppose we are given an integer point $x^I = (0, 0)^T$ and a value $0 < \delta < 0.5$. Suppose to distribute the δ value between the variables in two different ways, namely $x^A = (0, \delta)^T$ and $x^B = (\frac{\delta}{2}, \frac{\delta}{2})^T$. Consider now a suitable concave function such as*

$$\phi(t) = \min\big\{ \ln(t + \varepsilon), \ln[(1 - t) + \varepsilon] \big\}.$$

As we can see in Fig.1, we have

$$\psi(x^A) - \psi(x^I) = \phi(0) + \phi(\delta) - 2\phi(0) < 2\phi(\delta/2) - 2\phi(0) = \psi(x^B) - \psi(x^I),$$

or equivalently

$$\psi(x^A) < \psi(x^B).$$

More in general, given a point $x^I = (x_0^I, \ldots, x_n^I) \in \{0, 1\}^n$, a value $0 < \delta < 0.5$ and two points $x^A = (x_0^I, \ldots, x_i^I + \bar{\delta}, \ldots, x_n^I)$ and $x^B = (x_0^I + \frac{\bar{\delta}}{n}, \ldots, x_n^I + \frac{\bar{\delta}}{n})$ with

$$\bar{\delta} = \begin{cases} \delta & \text{if} \quad x_i^I = 0, \\ -\delta & \text{if} \quad x_i^I = 1, \end{cases}$$

choosing the concave function used in the Example 1

$$\phi(t) = \min\big\{ \ln(t + \varepsilon), \ln[(1 - t) + \varepsilon] \big\},$$

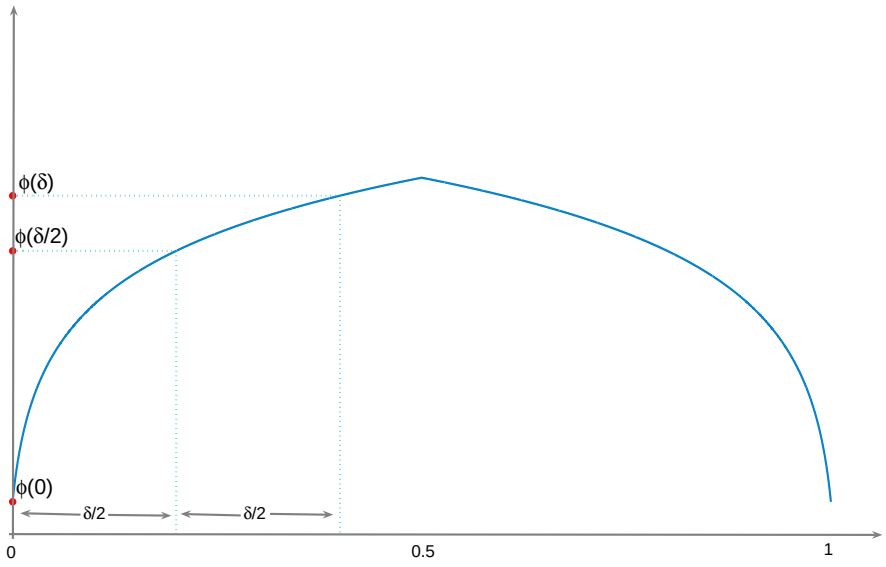

Figure 1: Behavior of the penalty term $\phi(t) = \min\big\{\ln(t+\varepsilon), \ln[(1-t)+\varepsilon]\big\}$.

we obtain

$$\psi(x^A) < \psi(x^B).$$

On the other hand, since the terms inside the ϕ function of the Feasibility Pump are linear we have

$$\psi(x^A) = \sum_{i\in I} \min\{x_i^A, 1-x_i^A\} = \sum_{i\in I} \min\{x_i^B, 1-x_i^B\} = \psi(x^B).$$

The following theoretical result gives us a guideline for choosing the nonlinear terms for the ϕ function:

Proposition 2 *Let $f : \mathbf{R} \to \mathbf{R}$ be a lower bounded concave function and let*

$$\hat{f} = \inf_{x\in\mathbf{R}} f(x)$$

be its lower bound. Then, given $x_i \in \mathbf{R}$, $i = 1, \ldots, n$

$$\sum_{i=1}^{n} (f(x_i) - \hat{f}) \geq f\left(\sum_{i=1}^{n} x_i\right) - \hat{f}. \tag{10}$$

Proof. See Appendix A. □

In this work, starting from Proposition 2 and the ideas developed in [23, 25], we replace the term (9) by the following concave nonsmooth terms:

Logarithmic function

$$\phi(t) = \min\big\{\ln(t+\varepsilon), \ln[(1-t)+\varepsilon]\big\} \tag{11}$$

Hyperbolic function

$$\phi(t) = \min\big\{-(t+\varepsilon)^{-p}, -[(1-t)+\varepsilon]^{-p}\big\} \tag{12}$$

Concave function

$$\phi(t) = \min\left\{1 - \exp(-\alpha t), 1 - \exp(-\alpha(1-t))\right\} \tag{13}$$

Logistic function

$$\phi(t) = \min\left\{[1 + \exp(-\alpha t)]^{-1}, [1 + \exp(-\alpha(1-t))]^{-1}\right\} \tag{14}$$

where $\varepsilon, \alpha, p > 0$. In Fig. 2, we compare the ϕ term related to the FP heuristic with those given by (11)-(14).

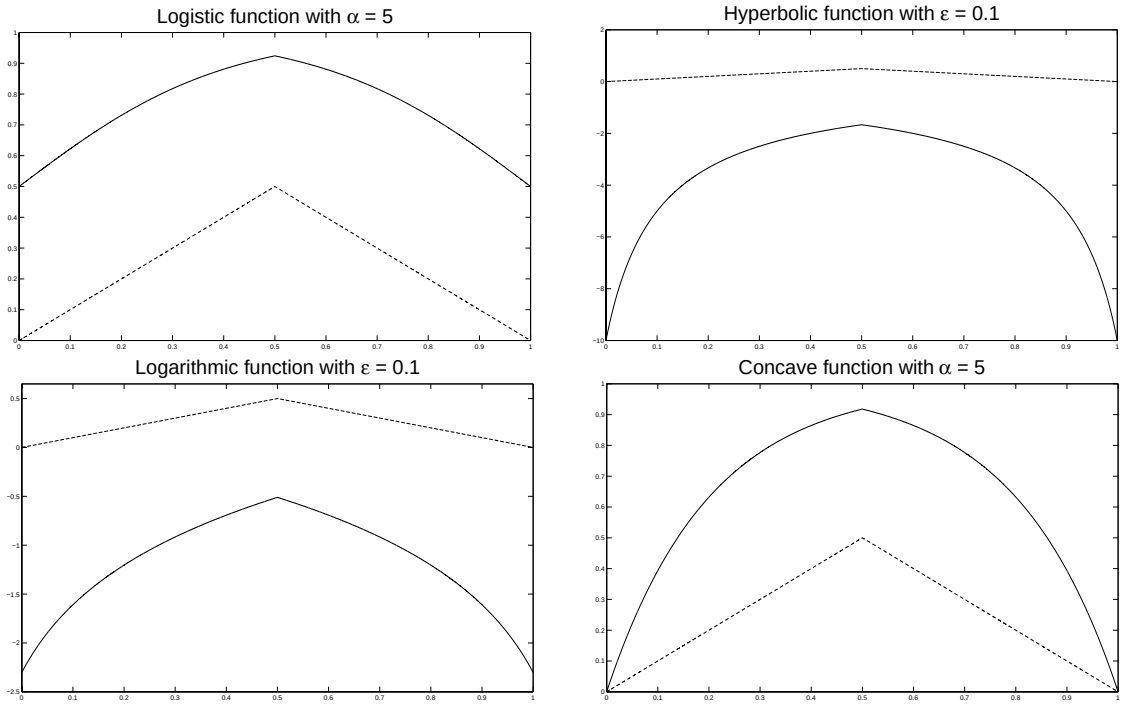

Figure 2: Comparison between the original FP term (dashed line) and the new terms (solid line).

For the merit functions described above it is possible to prove the following equivalence result:

Proposition 3 *Let f be a Lipschitz continuous function bounded on P. For every penalty term (11)-(14) a value $\bar{\varepsilon} > 0$ exists such that, for any $\varepsilon \in]0, \bar{\varepsilon}]$, problem*

$$\min f(x), \quad s.t. \quad x \in P, \;\; x_i \in \{0,1\}, \;\; \forall i \in I \tag{15}$$

and problem

$$\min f(x) + \tilde{\psi}(x, \varepsilon), \quad s.t. \quad x \in P, \;\; 0 \le x_i \le 1, \;\; \forall i \in I \tag{16}$$

where

$$\tilde{\psi}(x,\varepsilon) = \begin{cases} \psi(x) & \text{if } \psi \text{ is given by (8) and } \phi \text{ by (11)-(12)} \\ \dfrac{1}{\varepsilon}\,\psi(x) & \text{if } \psi \text{ is given by (8) and } \phi \text{ by (13)-(14)} \end{cases}$$

have the same minimum points.

Proof. See Appendix A. □

This theoretical result, although not strictly related to the aim of the present work, highlights the potentials of the merit functions (11)-(14). In fact, it indicates that the analysis carried out in this paper can be extended to an exact penalty approach for solving MIP problems.

Remark 2 *The penalty functions presented can be divided into two classes:*

- *Functions which tends to be unbounded from below as an inner penalty parameter goes to zero, namely those obtained by (11) and (12) terms.*
- *Functions that are always bounded between zero and one, namely those obtained by (13) and (14).*

The FP merit function belongs to the second class.

5 A reweighted version of the Feasibility Pump heuristic

The use of the merit functions (11)-(14) defined in the previous section leads to a new FP scheme in which the ℓ_1-norm used for calculating the next LP-feasible point is replaced with a "weighted" ℓ_1-norm of the form

$$\Delta_W(x, \tilde{x}) = \sum_{j \in I} w_j |x_j - \tilde{x}_j| = \|W(x - \tilde{x})\|_1, \tag{17}$$

where

$$W = diag(w_1, \ldots, w_n)$$

and w_j, $j = 1, \ldots, n$ are positive weights depending on the merit function ψ chosen. The main feature of the method is the use of an infeasibility measure that

- tries to discourage the optimal solution of the relaxation from being far from $\tilde{x}$ (similarly to the original FP algorithm);
- takes into account, in some way, the information carried by the LP-feasible points obtained at the previous iterations of the algorithm for speeding up the convergence to 0-1 feasible points.

Here we report an outline of the algorithm:

Reweighted Feasibility Pump (RFP) - basic version

Initialization: Set $k = 0$, let $\bar{x}^0 := \arg\min\{c^T x : Ax \geq b\}$
If ($\bar{x}^0$ is integer) return $\bar{x}^0$
Compute $\tilde{x}^0 = round(\bar{x}^0)$
While (not stopping condition) **do**

Step 1 Compute $\bar{x}^{k+1} := \arg\min\{\|W^k(x - \tilde{x}^k)\|_1 : Ax \geq b\}$

Step 2 If ($\bar{x}^{k+1}$ is integer) return $\bar{x}^{k+1}$

Step 3 Compute $\tilde{x}^{k+1} = round(\bar{x}^{k+1})$

Step 4 If (cycle detected) $\tilde{x}^{k+1} = perturb(\tilde{x}^k)$

Step 5 Update $k = k + 1$

End While

We assume that the *round* and *perturb* procedures are the same as those described in Section 2 for the original version of the FP heuristic. Anyway, different rounding and perturbing procedures can be suitably developed.
Following the same reasoning of Section 3, we can reinterpret the reweighted FP heuristic without perturbation as the unitary stepsize Frank-Wolfe algorithm applied to the merit function ψ. Let us now consider a generic iteration k of the reweighted FP. At Step 1, the algorithm computes the solution of the LP problem

$$\begin{aligned} \bar{x}^{k+1} \in \arg\min \Delta_{W^k}(x, \tilde{x}^k) \\ \text{s.t. } Ax \geq b \\ 0 \leq x_j \leq 1 \ \forall j \in I. \end{aligned} \tag{18}$$

Then, at Step 3, it rounds the solution $\bar{x}^k$, thus giving $\tilde{x}^{k+1}$.
Similarly to the FP algorithm, these two operations can be included in the unique step of calculating the solution of the following LP problem:

$$\begin{aligned} \min \sum_{j \in I: \bar{x}^k_j < \frac{1}{2}} w^k_j x_j - \sum_{j \in I: \bar{x}^k_j \geq \frac{1}{2}} w^k_j x_j \\ \text{s.t. } Ax \geq b \\ 0 \leq x_j \leq 1 \ \forall j \in I. \end{aligned} \tag{19}$$

By setting

$$w^k_j = |g^k_j|$$

with $g^k \in \partial\psi(\bar{x}^k)$, Problem (19) can be seen as the iteration of the Frank Wolfe method with unitary stepsize applied to the following minimization problem

$$\begin{aligned} \min \psi(x) \\ \text{s.t. } Ax \geq b \\ 0 \leq x_j \leq 1 \ \forall j \in I. \end{aligned} \tag{20}$$

Then the weighted distance can be interpreted as a way of "pumping" the integrality of $\tilde{x}^k$ into the new feasible point $\bar{x}^{k+1}$, while somehow trying to discourage the integer-constrained components of the LP-feasible point sequence $\{\bar{x}^k\}$ from changing their values once they get integer. In order to highlight the differences between the ℓ_1-norm and the weighted ℓ_1-norm we report the following example:

Example 2 *Consider the MILP problem:*

$$\begin{array}{ll} \min & c^T x \\ s.t. & x \in P \\ & x \in \{0,1\}^3 \end{array} \tag{21}$$

where $P \subset [0,1]^3$ is the polyhedron in Fig. 3. Let $x^L = \left(\frac{9}{20}, \frac{1}{8}, \frac{1}{8}\right)$ be the solution of the linear relaxation of (21) and $x^I = (0,0,0)$ be its rounding. The minimization of $\Delta(x, x^I) = \|x - x^I\|_1$

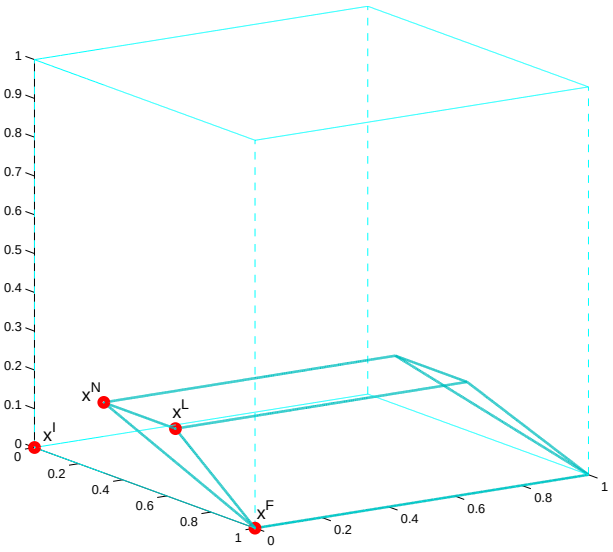

Figure 3: Feasible set of Problem 21.

over P leads to $x^N = \left(\frac{1}{8}, \frac{1}{8}, \frac{1}{8}\right)$, since $\|x^N - x^I\|_1 < \|x - x^I\|_1$, for all $x \in P$.
Consider now the weighted ℓ_1-norm obtained using the logarithmic merit function

$$\psi(x) = \sum_{i \in I} \min\left\{ \ln(x_i + \varepsilon), \ln[(1 - x_i) + \varepsilon] \right\},$$

where ε is a small positive value. By minimizing the weighted distance between x and x^I over P, we obtain the point $x^F = (1,0,0)$. In fact, we have

$$\Delta_W(x^F, x^I) < \Delta_W(x, x^I),$$

for all $x \in P$. Thus the ℓ_1-norm finds a solution which does not satisfy the integrality constraints, while the reweighted ℓ_1-norm gets an integer feasible solution.

6 Numerical Results

In this section we report computational results to compare our version of the FP with the Feasibility Pump algorithm described in [12]. The test set used in our numerical experience consists of 43 instances of 0-1 problems from MIPLIB [2]. All the algorithms were implemented in C and we have used ILOG Cplex [21] as solver of the linear programming problems. All tests have been run on an Intel Core2 E8500 system (3.16GHz) with 3.25GB of RAM.
We compare the FP with the reweighted version in three different scenarios:

1 **Randomly generated starting points**: for the terms (9), (11)-(14), we solved the corresponding penalty formulation (20) by means of the Frank-Wolfe algorithm using 100 randomly generated starting points. The aim of the experiment was to highlight the ability of each penalty formulation in finding an integer solution.

2 **FP vs RFP**: in order to evaluate the effectiveness of the new penalty functions, we compared the Feasibility Pump algorithm with the reweighted Feasibility Pump, in which the distance $\Delta_W(x, \tilde{x})$ is defined using the terms (11)-(14).

3 **FP vs Combined RFP**: we made a comparison between the Feasibility Pump algorithm and the reweighted Feasibility Pump with the distance $\Delta_W(x, \tilde{x})$ obtained combining two different penalty terms. The aim of the experiment was to show that the combination of two different functions can somehow improve the RFP algorithm performance.

We performed our experiments using:

- Penalty term (9) denoted by **FP**;
- Penalty term (11) denoted by **Log**, with $\varepsilon = 0.1$;
- Penalty term (12) denoted by **Hyp**, with $\varepsilon = 0.1$;
- Penalty term (13) denoted by **Conc**, with $\alpha = 0.5$;
- Penalty term (14) denoted by **Logis**, with $\alpha = 0.1$.

6.1 Computational results for randomly generated starting points

The results obtained on the 43 MIPLIB problems when using randomly generated starting points are shown in Table 1, where we report, for each method,the number of feasible points detected ($\sharp$ f.s.).
We can observe that the results obtained by means of the concave and the logistic functions, in terms of number of integer feasible solutions found, are comparable with those of the FP penalty function and slightly better than those obtained using the logarithmic and hyperbolic penalty functions. Anyway, the logarithmic and hyperbolic functions find, for a consistent number of

problems, the highest number of integer feasible solutions, so giving a good tool for finding an integer feasible solution.
This preliminary computational experience seems to confirm the validity of the new nonsmooth penalty functions in the search for integer feasible solutions over a polyhedral set and show that the functions here proposed can be a valid alternative to the FP penalty function. Furthermore, we remark that a wider availability of efficient penalty functions is important since it can ease the search of integer feasible solutions for different classes of problems.

6.2 Comparison between FP and RFP

The results of the comparison between the Feasibility Pump algorithm and the reweighted version obtained using the penalty terms (11)-(14) are shown in Tables 2 and 3. On the vertical axis of the tables, we have

- the number of iterations needed to find a solution (Iter),
- the objective function value of the first integer feasible solution found (Obj),
- the CPU time (Time).

We stop the algorithms if an integer solution is found or if the limit of 1500 iterations is reached. In case of failure, we report "-" for both Obj and Time. For four problems (*liu*, *mod011*, *modglob*, *opt1217*) the minimum-cost solution of the first LP relaxation is already an integer feasible solution, then no iterations of FP and RFP algorithms are performed. Since these problems are not meaningful for the comparison, we do not report the results in the tables.
By taking a look at the tables, we can notice that the RFP algorithm obtained using the **Conc** penalty (Conc-RFP algorithm) and the one obtained using the **Logis** penalty (Logis-RFP algorithm) are competitive with the FP in terms of both number of iterations and CPU time. They are also better than the RFP algorithm with the **Log** penalty (Log-RFP algorithm) and the one with the **Hyp** penalty (Hyp-RFP algorithm) that, in addition, have a larger number of failures. Despite these facts, Log-RFP and Hyp-RFP algorithms generally give good results in terms of objective function value. In particular, the Log-RFP and the Hyp-RFP respectively find the best objective function value for 15 and 17 instances over 43. The high number of failures might be due to the strong nonlinearity of the **Log** and **Hyp** functions, for which suitable perturbation heuristics should be developed.

6.3 Comparison between FP and combined RFP

As we have seen in Remark 2 the penalty functions can be divided into two classes:

(1) those generated by **Log** and **Hyp** terms;

(2) those generated by **FP**, **Conc** and **Logis** terms.

In this subsection, we show the effects of combining a function belonging to the first class, with any function of the other class.
Given two penalty functions $\psi_1(x)$ and $\psi_2(x)$, we can combine their effects by defining, at iteration k of the RFP algorithm, the following terms for the matrix W^k:

$$w_j^k = \lambda^k |g_j^k| + (1 - \lambda^k)|h_j^k| \quad j = 1, \ldots, n$$

Problem	FP	Log	Hyp	Conc	Logis
10teams	1	0	0	**3**	0
a1c1s1	0	0	0	0	0
aflow30a	0	0	**1**	0	0
aflow40b	**1**	0	0	0	0
air04	20	1	0	**25**	17
air05	41	8	0	**46**	43
cap6000	**26**	15	23	16	25
danoint	**2**	0	0	0	0
disktom	80	39	2	**92**	89
ds	3	0	0	5	**7**
fast0507	**100**	**100**	**100**	**100**	**100**
fiber	0	0	0	0	0
fixnet6	**10**	**10**	9	**10**	**10**
glass4	0	0	0	0	0
harp2	0	0	0	0	0
liu	0	0	0	0	0
markshare1	44	**63**	59	53	45
markshare2	53	54	**56**	52	53
mas74	**100**	**100**	**100**	**100**	**100**
mas76	**100**	**100**	**100**	**100**	**100**
misc07	1	4	**9**	0	2
mkc	0	0	0	0	0

Problem	FP	Log	Hyp	Conc	Logis
mod011	**100**	**100**	**100**	**100**	**100**
modglob	**100**	**100**	**100**	**100**	**100**
momentum1	0	0	0	**1**	**1**
net12	0	0	0	0	0
nsrand-ipx	**100**	**100**	**100**	**100**	**100**
nw04	42	11	3	**49**	43
opt1217	**100**	**100**	**100**	**100**	**100**
p2756	0	0	0	0	0
pk1	**100**	**100**	**100**	**100**	**100**
pp08aCUTS	83	**86**	**86**	84	84
pp08a	84	**86**	**86**	84	84
protfold	0	0	0	0	0
qiu	20	25	**30**	25	24
rd-rplusc-21	0	0	0	0	0
set1ch	0	0	0	**3**	**3**
seymour	**100**	**100**	**100**	**100**	**100**
sp97ar	22	**95**	**95**	**95**	**95**
swath	**16**	12	0	10	13
t1717	37	8	0	29	**44**
tr12-30	0	0	0	0	0
vpm2	0	0	0	0	0

Table 1: Comparison on MIPLIB problems. Randomly generated starting points

Problem	**FP**			**Log**, $\varepsilon = 0.1$			**Hyp**, $\varepsilon = 0.1$			**Conc**, $\alpha = 0.5$			**Logis**, $\alpha = 0.1$		
	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time
10teams	45	**992.0**	2.7	1500	-	-	1500	-	-	269	996.0	15.6	369	1040.0	22.5
a1c1s1	27	**19897.7**	1.8	19	24636.0	1.4	65	20243.2	4.4	65	24440.5	4.7	11	24153.9	1.3
aflow30a	8	5003.0	0.02	18	4071.0	0.02	11	**2712.0**	0.02	17	4457.0	0.03	7	4291.0	0.02
aflow40b	8	5010.0	0.05	10	3714.0	0.05	11	**3092.0**	0.05	6	4573.0	0.05	7	4703.0	0.03
air04	79	71891.0	184.8	1102	73677.0	1535.9	1500	-	-	7	59570.0	16.8	29	**70046.0**	77.2
air05	3	30342.0	6.9	7	30536.0	6.3	1500	-	-	3	29948.0	5.5	2	**29887.0**	4.4
cap6000	41	-1.00E6	1.4	11	-9.86E5	0.4	7	-1.30E6	0.3	11	-9.86E5	0.4	41	-1.00E6	1.4
danoint	4	77.0	0.1	50	80.5	0.8	41	**76.5**	0.7	28	80.5	0.6	4	78.0	0.1
disktom	7	**-5000.0**	3.4	5	**-5000.0**	1.8	1500	-	-	3	**-5000.0**	2.3	4	**-5000.0**	3.2
ds	443	1497.1	3847.2	1500	-	-	1500	-	-	464	1259.7	3452.1	214	**1133.5**	1829.4
fast0507	3	**181.0**	9.6	1	190.0	6.6	1	192.0	5.5	2	185.0	6.1	3	187.0	11.1
fiber	5	1.14E7	0.02	6	1.14E7	0.02	3	**1.33E6**	0.02	5	1.14E7	0.02	5	1.14E7	0.02
fixnet6	303	41897.0	0.3	203	**37422.0**	0.2	204	37825.0	0.2	203	37676.0	0.2	203	37772.0	0.2
glass4	31	4.00E9	0.03	13	1.30E10	0.02	255	9.73E9	0.4	195	8.43E9	0.1	151	**3.50E9**	0.1
harp2	315	-5.48E7	2.5	609	-4.46E7	4.9	711	-5.01E7	6.1	230	-4.22E7	1.8	16	**-5.85E7**	0.1
markshare1	1	**292.0**	0.0	1	**292.0**	0.0	1	**292.0**	0.0	1	**292.0**	0.0	1	**292.0**	0.0
markshare2	1	**160.0**	0.0	1	**160.0**	0.0	1	**160.0**	0.0	1	**160.0**	0.0	1	**160.0**	0.0
mas74	2	**14372.9**	0.0	1	19197.5	0.0	1	19197.5	0.0	1	19197.5	0.0	1	19197.5	0.02
mas76	2	**43774.3**	0.0	1	44877.4	0.0	1	44877.4	0.0	1	44877.4	0.0	1	44877.4	0.0

Table 2: Comparison on MIPLIB problems. FP vs RFP (Part I)

Problem	**FP**			**Log**, $\varepsilon = 0.1$			**Hyp**, $\varepsilon = 0.1$			**Conc**, $\alpha = 0.5$			**Logis**, $\alpha = 0.1$		
	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time
misc07	35	4730.0	0.05	43	**3665.0**	0.05	494	4300.0	0.5	51	4140.0	0.08	27	4480.0	0.05
mkc	3	-271.7	0.06	3	**-271.9**	0.06	3	**-271.9**	0.08	3	**-271.9**	0.06	3	-271.7	0.08
momentum1	1005	442824.9	156.7	661	423657.7	137.2	1500	-	-	307	443002.7	157.4	209	**423639.4**	114.7
net12	40	**337.0**	1.9	193	**337.0**	4.9	253	**337.0**	6.5	134	**337.0**	3.5	559	**337.0**	14.9
nsrand-ipx	4	377120.0	0.3	4	331360.0	0.3	5	**303360.0**	0.3	4	353440.0	0.3	4	367840.0	0.3
nw04	1	19882.0	0.4	117	33344.0	607.4	930	72548.0	5044.6	1	**19780.0**	0.4	1	19882.0	0.4
p2756$^{(*)}$	1500	-	-	1500	-	-	1500	-	-	1500	-	-	1500	-	-
pk1	1	**36.0**	0.0	1	**36.0**	0.0	1	**36.0**	0.0	1	**36.0**	0.0	1	**36.0**	0.0
pp08aCUTS	4	12590.0	0.02	4	12270.0	0.0	3	**10670.0**	0.0	4	11740.0	0.02	5	11670.0	0.0
pp08a	5	12570.0	0.0	3	**10810.0**	0.0	3	12210.0	0.0	5	11720.0	0.02	5	12680.0	0.02
protfold	363	-7.0	181.2	1500	-	-	1500	-	-	37	-7.0	9.0	210	**-15.0**	67.5
qiu	8	1276.5	0.2	5	**120.9**	0.1	9	1186.4	0.2	4	1504.6	0.1	4	1732.6	0.1
rd-rplusc-21$^{(*)}$	1500	-	-	1500	-	-	1500	-	-	1500	-	-	1500	-	-
set1ch	4	80149.0	0.0	4	78678.0	0.02	8	82402.5	0.02	3	**77964.5**	0.02	4	80274.5	0.02
seymour	4	**472.0**	1.3	3	482.0	1.2	2	495.0	1.1	3	**472.0**	1.3	3	**472.0**	1.3
sp97ar	7	1.36E9	1.8	4	**8.79E8**	1.4	4	1.75E10	1.1	7	1.35E9	1.6	7	1.21E9	1.5
swath	38	26807.9	1.6	25	**19148.8**	1.4	865	45947.9	32.4	79	47085.9	3.7	25	25846.9	1.3
t1717	40	240605.0	208.1	406	970384.0	2873.7	1500	-	-	23	**207584.0**	87.8	19	210529.0	93.1
tr12-30	28	287125.0	0.05	127	261987.0	0.2	70	**259743.0**	0.09	50	262722.0	0.08	61	261559.0	0.09
vpm2	3	17.8	0.0	2	**17.0**	0.02	2	**17.0**	0.0	3	17.8	0.0	3	17.8	0.0

Table 3: Comparison on MIPLIB problems. FP vs RFP (Part II)

with $\lambda^k \in [0,1]$, $g_j^k \in \partial\psi_1(\bar{x}^k)$ and $h_j^k \in \partial\psi_2(\bar{x}^k)$. We report the results obtained combining the following functions:

- **Fp** term and **Log**, denoted by **FP+Log**;
- **Conc** term and **Log** term, denoted by **Conc+Log**;
- **Logis** term and **Log** term, denoted by **Logis+Log**.

We set $\psi_1(x)$ equal to the penalty function obtained using the **Log** term and $\psi_2(x)$ equal to the other penalty function. We start with $\lambda^0 = 1$ and we reduce it every time a perturbation occurs. More precisely, we can have two different cases:

- *Weak Perturbation Update:* $\lambda^{k+1} = 0.5\lambda^k$
- *Strong Perturbation Update:* $\lambda^{k+1} = 0.1\lambda^k$

When a strong perturbation occurs, it means that the algorithm is stuck in a cycle. Then the updating rule significantly changes the penalty term, so moving towards the function belonging to the second class.
The results of the comparison between the Feasibility Pump algorithm and the combined RFP algorithm are shown in Tables 4 and 5. On the vertical axis of the tables, we have

- the number of iterations needed to find a solution (Iter),
- the objective function value of the first integer feasible solution found (Obj),
- the CPU time (Time).

We stop the algorithms if an integer solution is found or if the limit of 1500 iterations is reached. When a failure occurs, we report "-" for both Obj and Time. Also in this case, we do not report the results for problems: *liu*, *mod011*, *modglob*, *opt1217*.
As we can easily notice from the results reported in the tables, the combined RFP is competitive with the original version of the FP. Furthermore, the **Log** function combined with another penalty function generally gives better results than the **Log** function by itself.

6.4 Benchmarking Algorithms via Performance Profiles

In order to give a better interpretation of the results generated by the various algorithms we decided to use performance profiles [11]. We consider a set A of n_a algorithms, a set P of n_p problems and a performance measure $m_{p,a}$ (e.g. number of iteration, CPU time). We compare the performance on problem p by algorithm a with the best performance by any algorithm on this problem using the following *performance ratio*

$$r_{p,a} = \frac{m_{p,a}}{\min\{m_{p,a} \ : \ a \in A\}}.$$

Then, we obtain an overall assessment of the performance of the algorithm by defining the following value

$$\rho_a(\tau) = \frac{1}{n_p}\text{size}\{p \in P \ : \ r_{p,a} \leq \tau\},$$

Problem	**FP**			**FP +**	**Log**		**Conc +**	**Log**		**Logis +**	**Log**	
	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time	Iter	Obj	Time
10teams	45	992.0	2.7	24	**928.0**	1.0	35	954.0	1.6	40	1012.0	1.4
a1c1s1	27	**19897.7**	1.8	19	20140.4	1.5	87	32457.3	5.5	19	25560.4	1.4
aflow30a	8	5003.0	0.02	8	3504.0	0.02	8	**3048.0**	0.02	11	3926.0	0.02
aflow40b	8	5010.0	0.05	9	4320.0	0.05	14	6886.0	0.06	10	**3730.0**	0.05
air04	79	71891.0	184.8	33	70631.0	70.2	16	**65281.0**	37.1	19	68849.0	43.8
air05	3	**30342.0**	6.9	7	30536.0	6.5	7	30536.0	6.7	7	30536.0	6.8
cap6000	41	**-1.008E6**	1.4	41	**-1.008E6**	1.5	11	-9.860E5	0.4	41	**-1.008E6**	1.5
danoint	4	**77.0**	0.14	84	79.0	1.64	57	90.0	1.15	61	81.0	1.20
disktom	7	**-5000.0**	3.4	5	**-5000.0**	1.8	5	**-5000.0**	1.9	5	**-5000.0**	1.9
ds	443	1497.1	3847.2	861	**1430.9**	7283.2	550	1449.8	3977.4	468	1485.3	2740.2
fast0507	3	**181.0**	9.7	1	190.0	7.9	1	190.0	7.2	1	190.0	7.3
fiber	5	**1.145E7**	0.02	6	1.146E7	0.02	6	1.146E7	0.02	6	1.146E7	0.02
fixnet6	303	41897.0	0.344	203	37999.0	0.218	203	**37580.0**	0.219	203	38064.0	0.235
glass4	31	**4.00E9**	0.03	129	1.36E10	0.08	49	4.15E9	0.05	46	**4.00E9**	0.03
harp2	315	**-5.484E7**	2.5	44	-5.206E7	0.3	89	-4.633E7	0.6	26	-5.024E7	0.2
markshare1	1	**292.0**	0.0	1	**292.0**	0.0	1	**292.0**	0.0	1	**292.0**	0.0
markshare2	1	**160.0**	0.0	1	**160.0**	0.0	1	**160.0**	0.0	1	**160.0**	0.0
mas74	2	**14372.9**	0.0	1	19197.5	0.0	1	19197.5	0.0	1	19197.5	0.0
mas76	2	**43774.3**	0.0	1	44877.4	0.0	1	44877.4	0.0	1	44877.4	0.0

Table 4: Comparison on MIPLIB problems. FP vs combined RFP (Part I)

Problem	**FP** Iter	Obj	Time	**FP +** Iter	**Log** Obj	Time	**Conc +** Iter	**Log** Obj	Time	**Logis +** Iter	**Log** Obj	Time
misc07	35	4730.0	0.05	47	**4070.0**	0.08	226	**4070.0**	0.29	36	4605.0	0.05
mkc	3	-271.7	0.06	3	**-271.9**	0.08	3	**-271.9**	0.08	3	**-271.9**	0.08
momentum1	1005	442824.9	156.7	104	**410858.5**	39.6	1331	436727.5	161.9	613	430070.4	93.7
net12	40	**337.0**	1.9	131	**337.0**	3.4	616	**337.0**	16.2	148	**337.0**	5.7
nsrand-ipx	4	377120.0	0.3	4	**331360.0**	0.3	4	**331360.0**	0.4	5	343200.0	0.4
nw04	1	19882.0	0.4	10	40026.0	38.4	18	52746.0	54.6	7	**17814.0**	40.5
p2756$^{(*)}$	1500	-	-	1500	-	-	1500	-	-	1500	-	-
pk1	1	**36.0**	0.0	1	**36.0**	0.0	1	**36.0**	0.0	1	**36.0**	0.0
pp08aCUTS	4	12590.0	0.02	4	**11960.0**	0.02	4	**11960.0**	0.0	4	12270.0	0.0
pp08a	5	12570.0	0.0	3	**10930.0**	0.0	3	**10930.0**	0.0	3	**10930.0**	0.02
protfold	363	-7.0	181.2	211	-7.0	46.2	254	**-13.0**	45.2	176	-7.0	35.2
qiu	8	1276.5	0.16	5	**120.9**	0.11	5	**120.9**	0.11	5	**120.9**	0.11
rd-rplusc-21$^{(*)}$	1500	-	-	1500	-	-	1500	-	-	1500	-	-
set1ch	4	80149.0	0.0	4	**78678.0**	0.0	4	**78678.0**	0.02	4	**78678.0**	0.02
seymour	4	**472.0**	1.3	3	482.0	1.2	3	482.0	1.2	3	482.0	1.6
sp97ar	7	1.36E9	1.8	4	**8.79E8**	1.6	4	**8.79E8**	1.7	4	**8.79E8**	1.7
swath	38	**26807.9**	1.6	185	49581.2	6.9	130	48392.7	5.7	23	39427.1	1.1
t1717	40	240605.0	208.1	56	255065.0	287.5	35	218225.0	102.2	53	**216150.0**	185.1
tr12-30	28	287125.0	0.05	52	262689.0	0.08	158	261310.0	0.22	397	**255436.0**	0.56
vpm2	3	17.8	0.0	2	**17.0**	0.0	2	**17.0**	0.0	2	**17.0**	0.0

Table 5: Comparison on MIPLIB problems. FP vs combined RFP (Part II)

which represents the probability for algorithm $a \in A$ that the performance ratio $r_{p,a}$ is within a factor $\tau \in R$ of the best possible ratio. The function ρ_a represents the distribution function for the performance ratio. Thus $\rho_a(1)$ gives the fraction of problems for which the algorithm a was the most effective, $\rho_a(2)$ gives the fraction of problems for which the algorithm a is within a factor of 2 of the best algorithm, and so on. In Figures 4 and 5, we report the performance profiles related to the comparison between FP and RFP, in terms of number of iterations and CPU time, for $\tau \in [0, \bar{\tau}]$, with $\bar{\tau} = 2$ and $\bar{\tau} = 5$. From Figure 4 it is clear that Conc-RFP, Logis-RFP and Hyp-RFP functions have a higher number of wins with respect to FP both in terms of number of iterations and computational time. Furthermore, if we choose to be within a factor of 2, Conc-RFP and Logis-RFP are the best solvers and have a probability to solve a problem within a factor 2 greater than 80%. When we consider a factor of 5 (see Fig. 5), FP, Conc-RFP and Logis-RFP show similar performance.
In Figures 6 and 7, we report the performance profiles related to the comparison between FP and combined RFP, in terms of number of iterations and CPU time, for $\tau \in [0, \bar{\tau}]$, with $\bar{\tau} = 2$ and $\bar{\tau} = 5$. From Figure 6 we can notice that all the versions of the combined RFP algorithm have a higher number of wins with respect to FP both in terms of number of iterations and computational time. When we choose to be within a factor of 2, we have that

- Logis+Log-RFP show the best performance in terms of number of iterations;
- Logis+Log-RFP, Conc+Log-RFP and FP+Log-RFP are the best solvers in terms of CPU time (with a probability greater than 90%);

Finally, if we choose to be within a factor of 5 (see Fig. 7), the combined RFP versions are slightly worse than FP in terms of iterations, but show better CPU time performance.

7 Conclusions

In this paper, we started by interpreting the Feasibility Pump heuristic as a Frank-Wolfe method applied to a nonsmooth concave merit function. Then, we proposed new concave penalty functions that can be included in the FP scheme. Due to their nonlinear structure, these new functions should speed up the convergence towards integer feasible points. We reported computational results on a set of 43 difficult 0-1 MIP problems. The numerical experience we reported shows that the new version of the Feasibility Pump obtained by means of the proposed functions compares favorably with the original version of the FP. Apart from the improvement in efficiency, these experiments also highlight that the new merit functions are good alternatives to the FP function. As a final remark, we point out that a wider availability of efficient penalty functions is important since it can facilitate the search of integer feasible solutions for different classes of problems.
Future work will be devoted to the definition of new perturbing procedures suitable to the proposed functions, to the extension of our approach to MIP problems with general integer variables and to the development of new FP-like methods that, by taking into account the objective function values, guarantee the improvement of the solution quality.

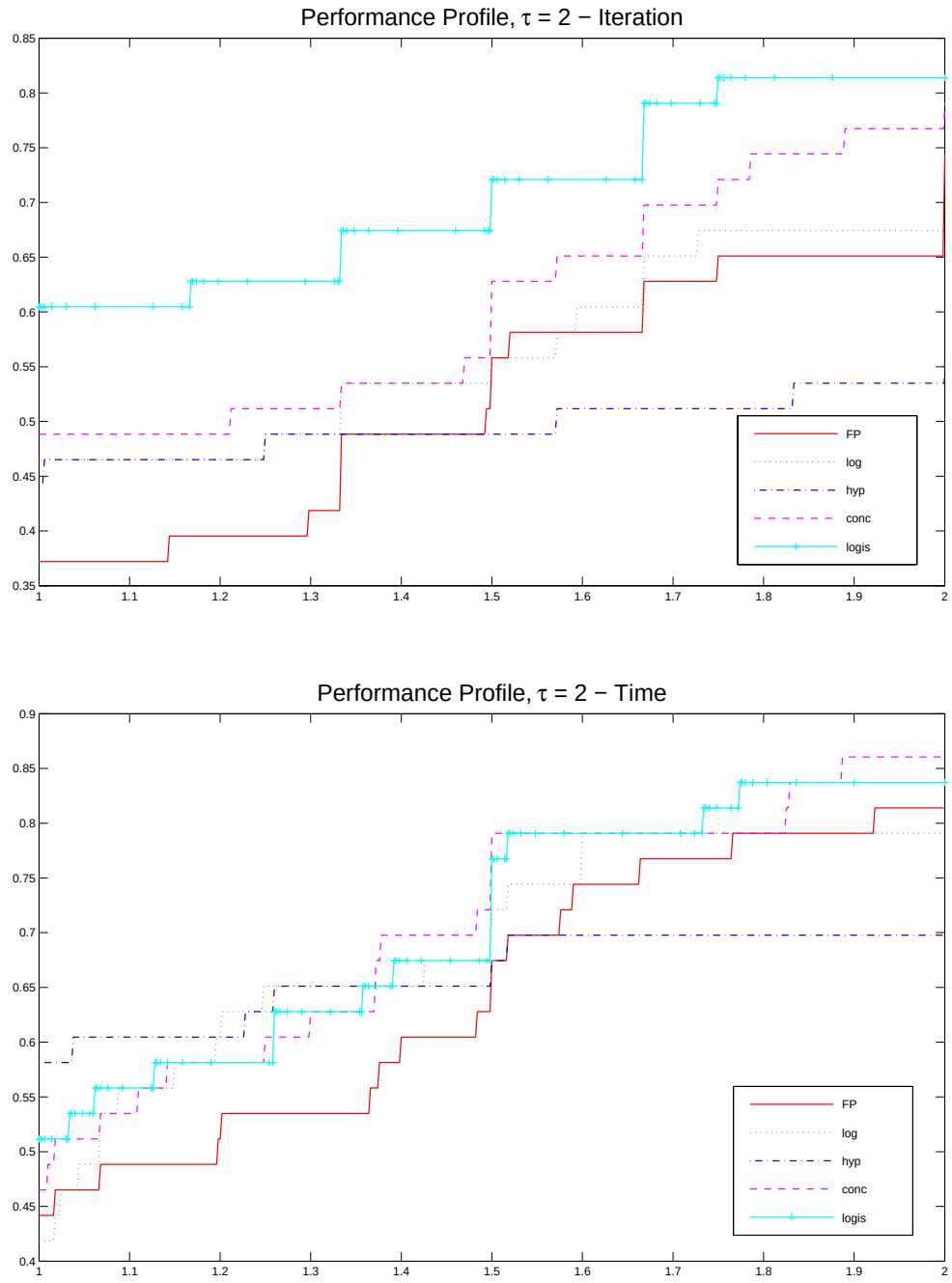

Figure 4: Comparison between FP and the different RFP versions. Performance profiles ($\tau = 2$)

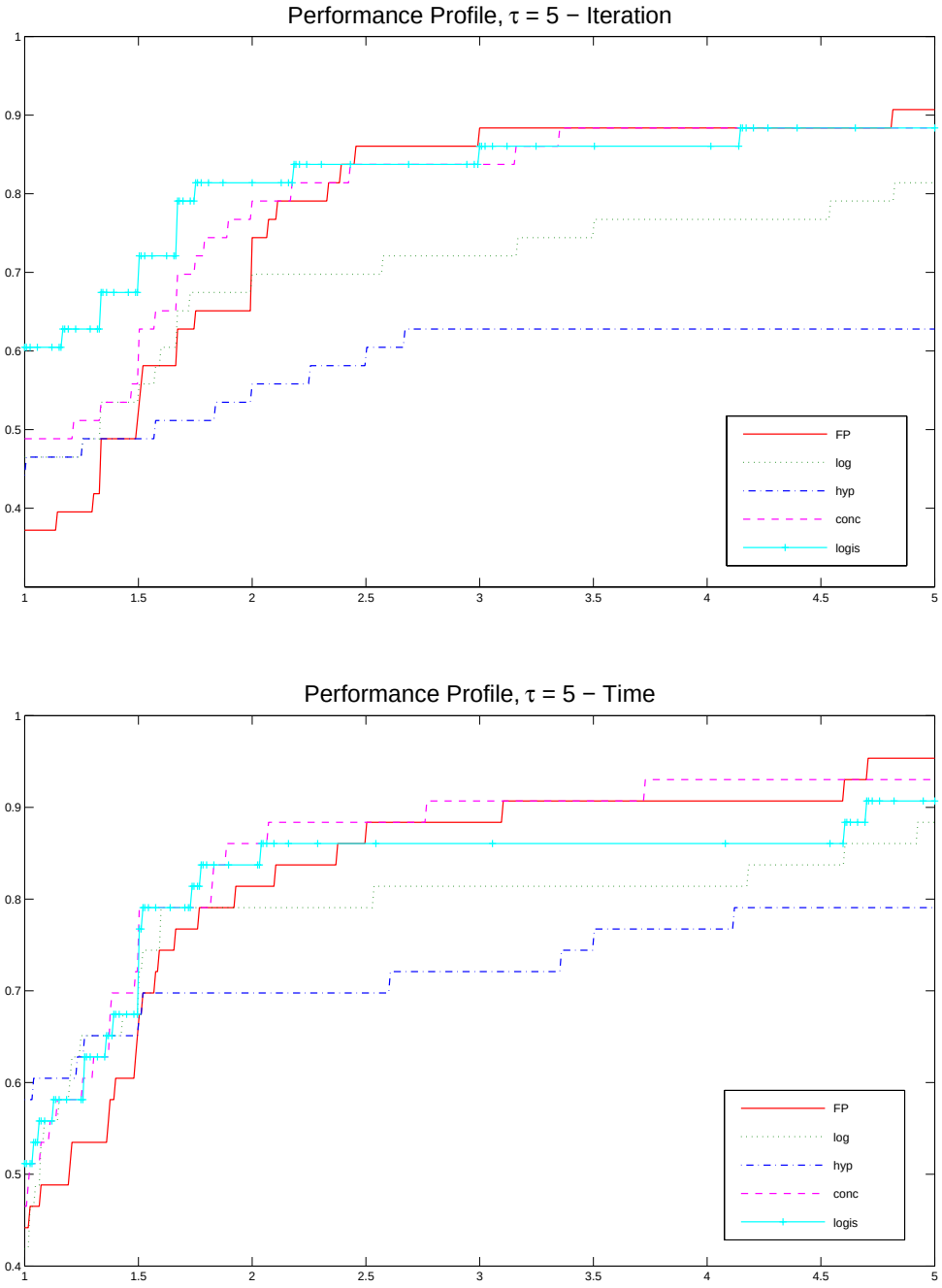

Figure 5: Comparison between FP and the different RFP versions. Performance profiles ($\tau = 5$)

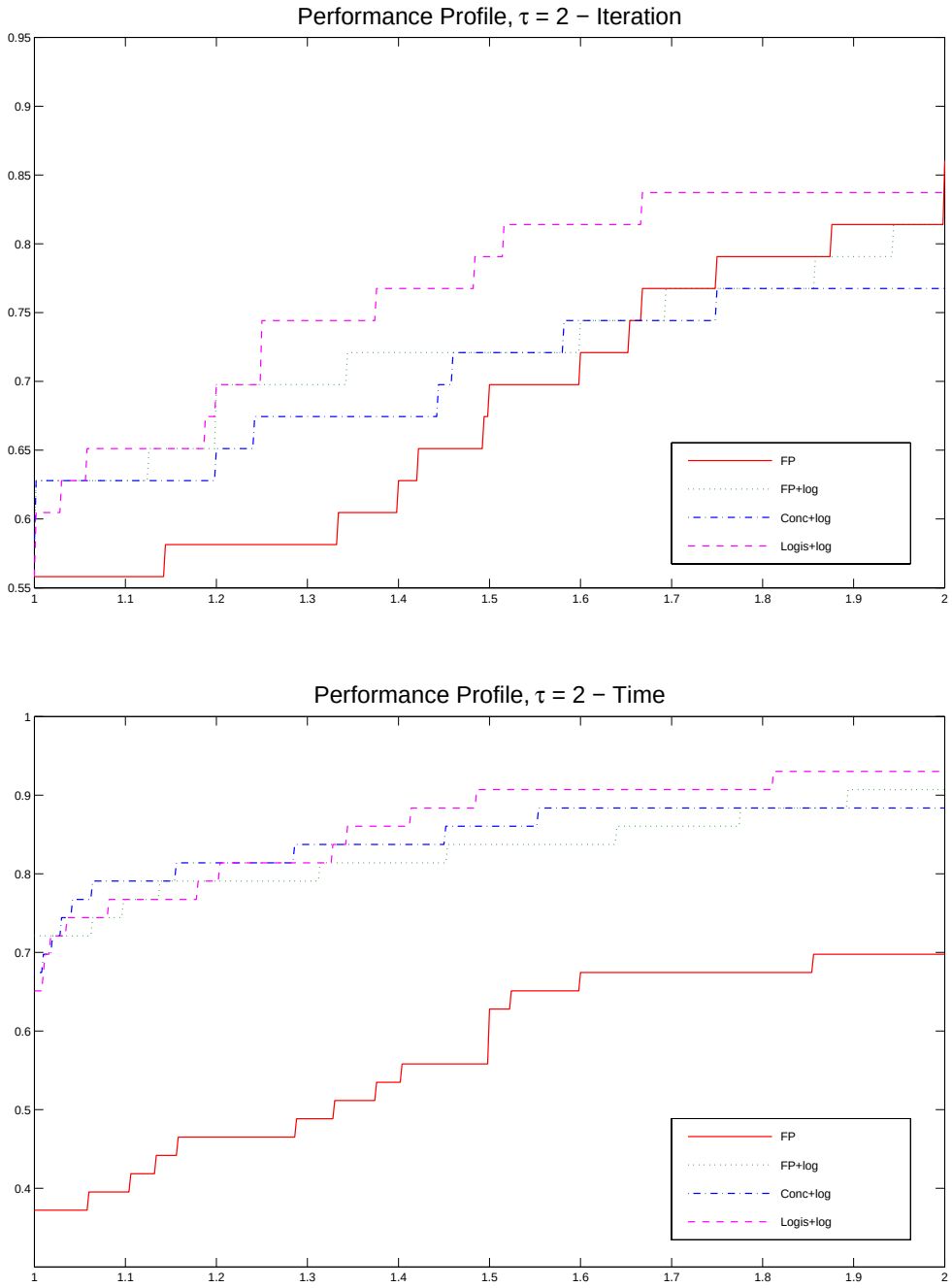

Figure 6: Comparison between FP and the different combined RFP versions. Performance profiles ($\tau = 2$)

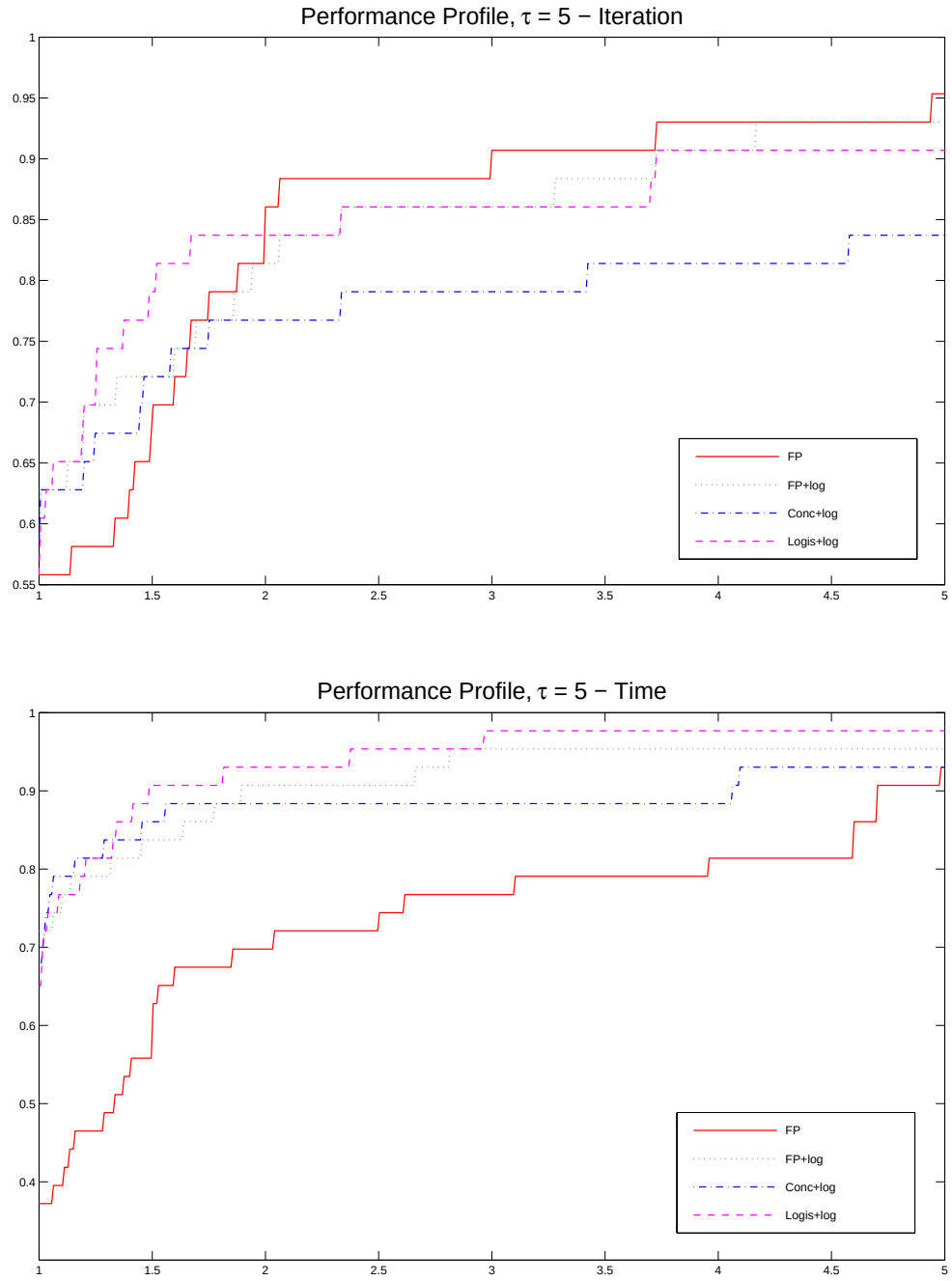

Figure 7: Comparison between FP and the different combined RFP versions. Performance profiles ($\tau = 5$)

8 Appendix A

For convenience of the reader we report the proofs of the theoretical results presented in the paper.

Here is the proof of Proposition 2.

Proof of Proposition 2. Let

$$x_1 = \theta y_1 + (1-\theta) z_1,$$
$$x_2 = \theta y_2 + (1-\theta) z_2$$

with $\theta \in [0,1]$ and y_1, z_1, y_2, z_2 as the following

$$y_1 = \sum_{i=1}^{n} x_i, \qquad z_1 = \frac{x_1 - \theta(\sum_{i=1}^{n} x_i)}{1-\theta},$$
$$y_2 = \frac{x_2 - (1-\theta)(\sum_{i=1}^{n} x_i)}{\theta}, \qquad z_2 = \sum_{i=1}^{n} x_i.$$

Since f is concave, we can write

$$\begin{aligned}
\sum_{i=1}^{n}(f(x_i) - \hat{f}) &= f(x_1) + f(x_2) + \sum_{i=3}^{n}(f(x_i) - \hat{f}) - 2\hat{f} \\
&= f(\theta y_1 + (1-\theta) z_1) + f(\theta y_2 + (1-\theta) z_2) + \sum_{i=3}^{n}(f(x_i) - \hat{f}) - 2\hat{f} \\
&\geq \theta f(y_1) + (1-\theta) f(z_2) + \sum_{i=3}^{n}(f(x_i) - \hat{f}) + (1-\theta) f(z_1) + \theta f(y_2) - 2\hat{f} \\
&= f\Big(\sum_{i=1}^{n} x_i\Big) + \sum_{i=3}^{n}(f(x_i) - \hat{f}) + (1-\theta)(f(z_1) - \hat{f}) + \theta(f(y_2) - \hat{f}) - \hat{f} \\
&\geq f\Big(\sum_{i=1}^{n} x_i\Big) - \hat{f}. \qquad (22)
\end{aligned}$$

The last inequality (22) follows since $f(x) - \hat{f} \geq 0$, $\forall x \in \mathbf{R}$. □

In order to prove Proposition 3, we recall a general result concerning the equivalence between an unspecified optimization problem and a parameterized family of problems.

Consider the problems

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in W \end{aligned} \qquad (23)$$

$$\begin{aligned} \min \quad & f(x) + \psi(x, \varepsilon) \\ \text{s.t.} \quad & x \in X \end{aligned} \qquad (24)$$

We state the following

Theorem 1 *Let W and X be compact sets. Let $\|\cdot\|$ be a suitably chosen norm. We make the following assumptions.*

A1) *The function f is bounded on X and there exists an open set $A \supset W$ and a real number $L > 0$, such that, $\forall\ x, y \in A$, f satisfies the following condition:*

$$|f(x) - f(y)| \leq L\|x - y\|. \tag{25}$$

The function ψ satisfies the following conditions:

A2) $\forall\ x, y \in W$ *and* $\forall\ \varepsilon \in \mathbb{R}_+$,

$$\psi(x, \varepsilon) = \psi(y, \varepsilon).$$

A3) *There exist a value $\hat{\varepsilon}$ and, $\forall\ z \in W$, there exists a neighborhood $S(z)$ such that, $\forall\ x \in S(z) \cap (X \setminus W)$, and $\varepsilon \in]0, \hat{\varepsilon}]$, we have*

$$\psi(x, \varepsilon) - \psi(z, \varepsilon) \geq \hat{L}\|x - z\|, \tag{26}$$

where $\hat{L} > L$ is chosen as in (25)*. Furthermore, let $S = \bigcup_{z \in W} S(z)$, $\exists\ \bar{x} \notin S$ such that*

$$\lim_{\varepsilon \to 0}[\psi(\bar{x}, \varepsilon) - \psi(z, \varepsilon)] = +\infty, \quad \forall\ z \in W, \tag{27}$$

$$\psi(x, \varepsilon) \geq \psi(\bar{x}, \varepsilon), \quad \forall\ x \in X \setminus S,\ \forall\ \varepsilon > 0. \tag{28}$$

Then, $\exists\ \tilde{\varepsilon} \in \mathbb{R}$ such that, $\forall\ \varepsilon \in]0, \tilde{\varepsilon}]$, Problems (23) *and* (24) *have the same minimum points.*

Proof. See [23].

Now we give the proof of the Proposition 3, with

$$W = \Big\{x \in P : x_i \in \{0, 1\},\ \forall i \in I\Big\}, \quad X = \Big\{x \in P : 0 \leq x_i \leq 1,\ \forall i \in I\Big\}.$$

Proof of Proposition 3. As we assume that the function f satisfies assumption $A1$) of Theorem 1, the proof can be derived by showing that every penalty term (11)-(14) satisfies assumption $A2$) and $A3$) of Theorem 1.

Consider the penalty term (11).
Let c be the cardinality of I, for any $x \in W$ we have

$$\psi(x, \varepsilon) = c \cdot \log(\varepsilon)$$

and $A2$) is satisfied.
We now study the behavior of the function $\phi(x_i)$, $i \in I$, in a neighborhood of a point $z_i \in \{0, 1\}$. We distinguish three different cases:

1. $z_i = 0$ and $0 < x_i < \rho$ with $\rho < \frac{1}{2}$: We have that $\phi(x_i) = \ln(x_i + \varepsilon)$ which is continuous and differentiable for $0 < x_i < \rho$, so we can use mean value Theorem obtaining that

$$\phi(x_i) - \phi(z_i) = \left(\frac{1}{\tilde{x}_i + \varepsilon}\right)|x_i - z_i|, \tag{29}$$

with $\tilde{x}_i \in (0, x_i)$. Since $\tilde{x}_i < \rho$, we have

$$\phi(x_i) - \phi(z_i) \geq \left(\frac{1}{\rho + \varepsilon}\right)|x_i - z_i|. \tag{30}$$

Choosing ρ and ε such that

$$\rho + \varepsilon \leq \frac{1}{\hat{L}}, \tag{31}$$

we obtain

$$\phi(x_i) - \phi(z_i) \geq \hat{L}|x_i - z_i|. \tag{32}$$

2. $z_i = 1$ and $1 - \rho < x_i < 1$ with $\rho < \frac{1}{2}$: We have that $\phi(x_i) = \ln(1 - x_i + \varepsilon)$ which is continuous and differentiable for $1 - \rho < x_i < 1$, so we can use mean value Theorem obtaining that

$$\phi(x_i) - \phi(z_i) = \left(-\frac{1}{1 - \tilde{x}_i + \varepsilon}\right)(x_i - z_i) = \left(\frac{1}{1 - \tilde{x}_i + \varepsilon}\right)|x_i - z_i|, \tag{33}$$

with $\tilde{x}_i \in (x_i, 1)$. Since $\rho < \frac{1}{2}$ and $\tilde{x}_i > 1 - \rho$ we have $\frac{1}{1-\tilde{x}_i} > \frac{1}{\rho}$ then

$$\phi(x_i) - \phi(z_i) \geq \left(\frac{1}{\rho + \varepsilon}\right)|x_i - z_i|. \tag{34}$$

We have again that (32) holds when ρ and ε satisfy (31).

3. $z_i = x_i = 0$ or $z_i = x_i = 1$: We have $\phi(x_i) - \phi(z_i) = 0$.

We can conclude that, when ρ and ε satisfy (31),

$$\psi(x, \varepsilon) - \psi(z, \varepsilon) \geq \hat{L} \sum_{i \in I} |x_i - z_i| \geq \hat{L} \sup_{i \in I} |x_i - z_i| \tag{35}$$

for all $z \in W$ and all x such that $\sup_{i \in I} |x_i - z_i| < \rho$.
Now we define $S(z) = \{x \in \mathbf{R}^n : \sup_{i \in I} |x_i - z_i| < \rho\}$ and $S = \bigcup_{i=1}^{N} S(z_i)$ where N is the number of points $z \in W$.
Let $\bar{x} \notin S$ be such that $\exists j \in I : \ \bar{x}_j = \rho$ ($\bar{x}_j = 1 - \rho$) and $\bar{x}_i \in \{0, 1\}$ for all $i \neq j$, $i \in I$.
Let $\{\varepsilon^k\}$ be a sequence such that $\varepsilon^k \to 0$ for $k \to \infty$, we can write for each $z \in W$:

$$\begin{aligned} \lim_{k \to \infty} [\psi(\bar{x}, \varepsilon^k) - \psi(z, \varepsilon^k)] &= \lim_{k \to \infty} \Big([\ln(\rho + \varepsilon^k) + (c-1)\ln(\varepsilon^k)] - c\ln(\varepsilon^k) \Big) = \\ &\lim_{k \to \infty} \Big(\ln(\rho + \varepsilon^k) - \ln(\varepsilon^k) \Big) = +\infty \end{aligned}$$

and (27) holds.
Then $\forall x \in X \backslash S$, and $\forall \varepsilon > 0$ we have for the monotonicity of the logarithm:

$$\begin{aligned} \psi(x, \varepsilon) - \psi(\bar{x}, \varepsilon) &= \sum_{i \neq j} \min\{\ln(x_i + \varepsilon), \ln(1 - x_i + \varepsilon)\} - (c-1)\ln(\varepsilon) \\ &+ \min\{\ln(x_{\tilde{\jmath}} + \varepsilon), \ln(1 - x_{\tilde{\jmath}} + \varepsilon)\} - \ln(\rho + \varepsilon) \geq 0, \end{aligned}$$

where $\rho \leq x_{\tilde{\jmath}} \leq 1 - \rho$. Then (28) holds, and Assumption $A3$) is satisfied.
The proofs of the equivalence between (15) and (16) using the other penalty terms follow by repeating the same arguments used here. □

References

[1] T. Achterberg and T. Berthold. *Improving the feasibility pump.* Discrete Optimization, 4, pp 77–86, 2007.

[2] T. Achterberg, T. Koch, and A. Martin. *MIPLIB 2003.* Operations Research Letters, 34, pp 361–372, 2006. Problems available at http://miplib.zib.de.

[3] E. Balas, S. Ceria, M. Dawande, F. Margot, G. Pataki. *OCTANE: A new heuristic for pure 0-1 programs.* Operations Research, 49(2), pp 207-225, 2001.

[4] E. Balas, C.H. Martin. *Pivot-and-complement: A heuristic for 0-1 programming.* Management Science, 26(1), pp 86-96, 1980.

[5] E. Balas, S. Schmieta, C. Wallace.*Pivot and shifta mixed integer programming heuristic.* Discrete Optimization, 1(1), pp 3-12, 2004.

[6] L. Bertacco, M. Fischetti, and A. Lodi. *A feasibility pump heuristic for general mixed-integer problems.* Discrete Optimization, 4, pp 63–76, 2007.

[7] J. Eckstein and M. Nediak. *Pivot, cut, and dive: a heuristic for 0-1 mixed integer programming.* Journal of Heuristics, 13, pp 471–503, 2007.

[8] P. Bonami, L.T. Biegler, A.R. Conn, G. Cornuejols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N.Sawaya and A. Waechter. *An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs.* IBM Research Report RC23771, To appear in Discrete Optimization, in press.

[9] P. Bonami, G. Cornuejols, and A. Lodi, F. Margot. *A feasibility pump for mixed integer nonlinear programs.* Mathematical Programming, 119, pp 331–352, 2009.

[10] E. Danna, E. Rothberg, C. Le Pape. *Exploring relation induced neighborhoods to improve MIP solution.* Mathematical Programming 102, 1, pp 71–90, 2005.

[11] E. D. Dolan, J. J. Moré. *Benchmarking optimization software with performance profile.* Mathematical Programming 91, pp 201–213, 2002.

[12] M. Fischetti, F. Glover, A. Lodi. *The Feasibility Pump.* Mathematical Programming, 104, pp 91–104, 2005.

[13] M. Fischetti, A. Lodi. *Local Branching.* Mathematical Programming, 98(1-3), pp 23–47, 2003.

[14] M. Fischetti, D. Salvagnin. *Feasibility pump 2.0.* Mathematical Programming Computation, 1, pp 201–222, 2009.

[15] F. Glover, M. Laguna. *General purpose heuristics for integer programming part I.* Journal of Heuristics, 3, 1997.

[16] F. Glover, M. Laguna. *General purpose heuristics for integer programming part II.* Journal of Heuristics, 3, 1997.

[17] F. Glover, M. Laguna. *Tabu Search.* Kluwer Academic Publisher, Boston, Dordrecht, London, 1997.

[18] F. Glover, A. Løkketangen, D.L. Woodruff. *Scatter search to generate diverse MIP solutions.* in: M. Laguna, J. Gonzàlez-Velarde (Eds.), OR Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research, Kluwer Academic Publishers, pp. 299317, 2000.

[19] F.S. Hillier, *Efficient heuristic procedures for integer linear programming with an interior.* Operations Research, 17, pp 600-637, 1969.

[20] F.S. Hillier, R.M. Saltzman., *A heuristic ceiling point algorithm for general integer linear programming.* Management Science, 38(2), pp 263-283, 1992.

[21] ILOG, Cplex. http://www.ilog.com/products/cplex.

[22] A. Løkketangen, F. Glover. , *Solving zero/one mixed integer programming problems using tabu search.* European Journal of Operations Research, 106, pp 624-658, 1998.

[23] S. Lucidi, F. Rinaldi. *Exact penalty functions for nonlinear integer programming problems.* Accepted for publication on JOTA.

[24] O. L. Mangasarian. *Solutions of General Linear Complementarity Problems via Nondifferentiable Concave Minimization.* Acta Mathematica Vietnamica, 22(1), pp 199–205, 1997.

[25] Rinaldi F. *New results on the equivalence between zero-one programming and continuous concave programming*, Optimization Letters, Vol. 3, No. 3, 377–386 (2009).